G-Repository is an intelligent storage environment that manages the synchronization of changes made to Test Data Projects developed on GenRocket’s synthetic test data automation platform. It ensures all synthetic data generated by the system is accurate and fully up-to-date at all times.

G-Repository provides the GenRocket Organization Administrator (Org Admin) with visibility and control over the deployment of the platform as system usage scales across users and teams located around the world. G-Repository monitors changes made in GenRocket Cloud and communicates them to all GenRocket Runtime instances deployed locally on the customer’s on-premise environment or in the customer’s secure virtual private cloud.

G-Repository is a set of Java Jars installed by the Org Admin on any system hardware designated for generating synthetic test data and deployed on existing user machines or a dedicated device as determined by the Org Admin.
In today’s Agile/DevOps environment, applications and databases are constantly changing. Test data used to validate these applications must dynamically adapt to these changes. That’s because the quality of the test is only as good as the quality of the test data used for testing.

GenRocket has incorporated synthetic data lifecycle management into all aspects of its Test Data Automation platform to ensure the highest quality synthetic data is used for testing. There are four stages of the synthetic data lifecycle and each one serves an important purpose:

1. MODEL the structures and relationships of the target environment
2. DESIGN the variety and volume of synthetic data need for testing
3. DEPLOY synthetically generated test data into your test environment
4. MANAGE multiple releases of test data projects with version control

Lifecycle management requires intelligent automation to detect & synchronize changes throughout this lifecycle.
G-Repository uses intelligent automation to ensure that changes made anywhere in the GenRocket environment are synchronized everywhere those changes impact Test Data Projects. Generated test data must always be accurate and up-to-date.

Synchronized changes are made behind the scenes and under the hood so testers can focus on designing and deploying the synthetic data needed for testing. G-Repository is always running to perform the essential steps required for synchronized changes.

- **Detect** data model changes in the data environment
- **Update** components impacted by data model changes
- **Monitor** configuration changes to Test Data Projects
- **Refactor** all impacted Test Data Project components
- **Synchronize** the projects used by all team members
- **Log** system utilization data for analytics & reporting

These steps are performed continuously and transparently as testers, data architects and developers work together to automate the generation of synthetic test data.
**G-Repository Architecture**

**GenRocket Cloud**
- **GenRocket Cloud Scenario Download Service**
  - Responsible for downloading all GenRocket components
- **GenRocket Cloud Stats Upload Service**
  - Responsible for uploading statistics after each Scenario run
- **GenRocket Cloud Licensing Service**
  - Responsible for license checking before running a Scenario
- **GenRocket Cloud G-Repository Server**
  - Responsible for managing GenRocket repositories within a secure customer environment

**Secure Customer Environment**
- **G-Repository Client**
  - Responsible for managing GenRocket repositories on a user’s computer or test server

**Diagram Notes**
- "<Linux Server>> GenRocket Cloud Scenario Download Server"
- "<Linux Server>> GenRocket Cloud Stats Upload Server"
- "<Linux Server>> GenRocket Cloud License Check Server"
- "<PC>> Client 1, 2, 3, 4, 5, N"
- "<Client G-Repository Server (GRS)>
- "<uses>
- "https"
The architecture of G-Repository is highly modular and incorporates several streamlined, secure and scalable system services.

**G-Repository Server** is the main software module and is responsible for managing GenRocket repositories within a secure customer environment. **G-Repository server is not hardware and can execute on any physical or virtual machine.** It can be installed on either a shared or dedicated system.

**G-Repository Client** is the software module responsible for managing GenRocket repositories on a user’s computer or test server. It communicates directly with the G-Repository Server module to receive updates for test data projects.

**GenRocket Cloud Scenario Download Service** is responsible for downloading all GenRocket project components that are designed or revised in the GenRocket Cloud.

**GenRocket Cloud Licensing Service** is responsible for license checking before running Scenarios in a Test Data Project.

**GenRocket Cloud Stats Upload Service** is responsible for uploading operational statistics after each Scenario run to aggregate usage data for system analytics and reporting.
How G-Repository Works

G-Repository software modules work together to perform a continuous synchronization process.

Here is a functional overview of its fully automated process flow:

1. Monitor recent actions performed on GenRocket Cloud
2. Mirror recent actions performed on GenRocket Cloud in its local repository
3. Monitor run requests from multiple G-Repository Clients simultaneously
   - Perform license checks as a secure proxy for G-Repository Clients
   - Check if G-Repository Client needs updating
   - Send Updates to G-Repository Client
   - Collect G-Repository Client Scenario-run statistics
4. Send G-Repository Client Scenario-run statistics to GenRocket Cloud

GenRocket’s use of intelligent automation keeps the entire GenRocket test data automation platform fully up-to-date and in-sync at all times.
G-Repository is the engine that powers **synthetic data lifecycle management** across your organization. It controls a suite of intelligent automation features and orchestrates their behavior during each lifecycle stage.

**GenRocket lifecycle management features automate the detection, synchronization and version control over data structures and design patterns.**

**XTS** (Extract Table Schema) ensures data relationships and data attributes accurately reflect data model changes.

**G-Delta** monitors changes between database metadata and a project whose *Domains* and *Attributes* represent the data model.

**G-Refactor** auto-updates *Domains*, *Attributes* and *Scenarios* whenever the *Template Domain* or *Domain Relationships* are changed.

**Project Versioning** ensures alignment with a given software release and allows testers to repurpose a project for another application.

**G-Analytics** provides utilization data and system reports for optimizing the global deployment of the GenRocket platform across all users and teams.
The Importance of G-Repository

It's critically important that Org Admins install G-Repository in order to realize the full benefits of Synthetic Data Lifecycle Management.

- Streamlined system operations and fewer manual procedures
- A foundation for flexible and scalable enterprise-wide deployment
- Assured accuracy for all synthetic test data generation operations
- More effective support of GenRocket software by customer service
- Full compatibility with new releases & full access to new features

Install G-Repository by following an easy installation process:

1. **G-Repository Server** is a set of Java JARs installed on a centrally accessible machine in the secure environment

2. **G-Repository Client** is a set of Java JARs installed on a Client Machine (Developer, Test Engineer, Tester)

It's an easy process that is fully documented in our Knowledge Base and in our Flight School on-demand learning environment.